Gitin lataus**: git-scm.com/downloads**

Asennetun version tarkastus: **git –version**

Käyttäjän konfigurointi:

**git config –global user.name “etunimi sukunimi”**

**git config –global user.email** [**user@mail.com**](mailto:user@mail.com)//näkyy commiteissa, **git log**

**git init** initialisoi kansion gitin käyttöön ja luo .git -piilokansion

**git status** näyttää missä branchissa ollaan, ja kertoo lisätyt/muutetut/poistetut tiedostot

sekä mitkä tiedostot ovat staging areassa valmiita ”commitettavaksi” yms.

**git add** **file1 file2** lisäisi file1 ja file2:n staging areaan

**git add .**  lisäisi kaikki tiedostot staging areaan

**git commit -m ”add index.html”** tekee commitin, ja commit message:ksi tulee add index.html

**git log** näyttää tehdyt commitit. perään voi laittaa **–oneline**, tulee pelkistetty näkymä

**git checkout --*filename*** palauttaa tiedoston edelliseen tilaan

**git checkout *commit-hash*** siirtyy toiseen, haluttuun commit:iin. HEAD siirtyy, mitään ei poisteta

**git checkout master** tai suositeltu uudempi **git switch master** siirtyy master-branchiin

**git checkout *haaran\_nimi*** tai **git switch *haaran\_nimi*** siirtyy toiseen haaraan

**.gitignore** -tiedostoon voi määrittää tiedostot /tiedostoformaatit jotka git tulee jättämään

huomiotta (nämä eivät tule committeihin eikä git ilmoita jos niihin tehdään muutoksia)

**git branch** näyttää nykyisen haaran sekä muut haarat. nykyinen näytetään esim. \*master

**git branch *haaran\_nimi*** luo branchin(eli haaran), kopioi sisällön uuteen

**git merge *haaran\_nimi***  yrittää mergetä haaran toiseen(eli tehdä uuden commitin jossa

yhdistää sillä hetkellä olevaan haaraan *haaran\_nimi*-haaran)

**git merge –abort** =undo merge

**git reset –hard *commit\_hash*** siirtää masterin haluttuun committiin,VAARALLINEN, jos

menee historiassa taaksepäin, niin poistaa tiedostoja!

Huom! merget eivät muuta eivätkä poista olemassa olevia tiedostoja

**git stash** (“stashaa”, eli “jemmaa”) tiedostot, jotka ovat WIP eli work in progress

**git stash –u**  myös untracked files( -u=untracked)

**git stash list** näyttää kaikki stashit

**git stash apply 0** (0:n paikalle indeksi, mikä halutaan napata jemmasta, 0 on aina uusin, 1 toiseksi uusin jne.)

**git stash show 0**  näyttää tiedostot jotka ovat jemmassa (sivussa)

**git stash show –p 0**  -p näyttää lisäksi vielä tiedostojen muutetun sisällön (-p = patched)

**git stash push –m “updated files”**  push –m “teksti” voidaan antaa stashille nimi

**git stash branch new\_branch\_name 0** luo uuden branchin halutusta stash-indeksistä ja siirtyy siihen

**git stash pop**  hakee uusimman stashin working directoryyn ja poistaa(“pop out”) sen stashista.

**git stash clear**  tyhjentää kaikki stashit eli poistaa kaikki

**git stash drop 0**  poistaa yksittäisen stashin indeksin mukaan